PATENT ABSTRACTS OF JAPAN

(11)Publication number : 09-097203
(43)Date of publication of application : 08.04.1997

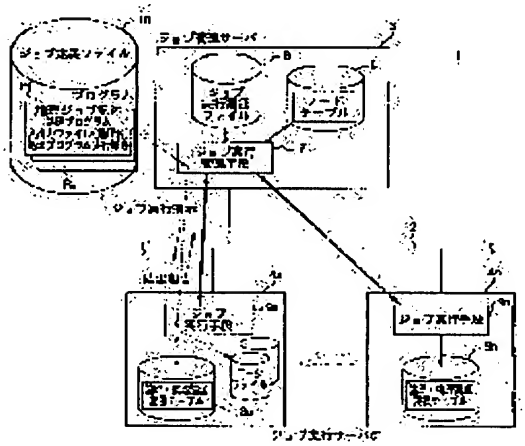
(51)Int.Cl. G06F 12/00
G06F 9/46
G06F 11/34
G06F 15/16

(21)Application number : 07-253076
(22)Date of filing : 29.09.1995

(71)Applicant : NRI & NCC CO LTD
(72)Inventor : YAMAMOTO SOICHI
UENO AYUMI
MUTO YOSHITO
YOSHIDA NOBUKATSU
INADA YOICHI
YAMAWAKI MASANORI
FUJISHIMA MIHO

(54) DISTRIBUTED PROCESSING SYSTEM

(57)Abstract:
PROBLEM TO BE SOLVED: To provide a distributed processing system with which the descriptions of programs by systematically unified logical file titles are permitted and a uniform job execution history can be outputted.
SOLUTION: This system is provided with a job managing server 3 for managing entire distributed processing and job executing servers 4a...4n for executing the jobs of programs and the programs to be distributedly processed are described by processing programs, the execution order of processing programs and logical title files. Then, the job managing server 3 is provided with a node table 5 recording the names or the like of job executing servers, a job execution history file 6 and a job execution managing means 7 for managing the execution of jobs, and the job executing servers 4a～4n are provided with logical/physical resource converting tables 8a...8n for converting the logical title files into physical resource files in reading and job executing means 9a...9n for executing jobs.



LEGAL STATUS

[Date of request for examination] 19.03.2002
[Date of sending the examiner's decision of rejection]
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]
[Date of final disposal for application]
[Patent number] 3795107
[Date of registration] 21.04.2006
[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

引用文献3

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平9－97203

(43)公開日　平成９年(1997)４月８日

(51)Int.Cl.[6]		識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｇ０６Ｆ	12/00	５４５		Ｇ０６Ｆ	12/00	５４５Ｂ
	9/46	３６０			9/46	３６０Ｂ
	11/34		7313－5B		11/34	Ｃ
	15/16				15/16	４２０Ｊ

審査請求　未請求　請求項の数２　ＯＬ　（全 11 頁）

(21)出願番号　特願平7－253076

(22)出願日　平成７年(1995)９月29日

(71)出願人　000155469
株式会社野村総合研究所
東京都中央区日本橋１丁目10番１号

(72)発明者　山　本　宗　一
神奈川県横浜市保土ケ谷区神戸町134番地
株式会社野村総合研究所内

(72)発明者　上　野　歩
神奈川県横浜市保土ケ谷区神戸町134番地
株式会社野村総合研究所内

(72)発明者　武　藤　義　人
神奈川県横浜市保土ケ谷区神戸町134番地
株式会社野村総合研究所内

(74)代理人　弁理士　佐藤　一雄　（外３名）

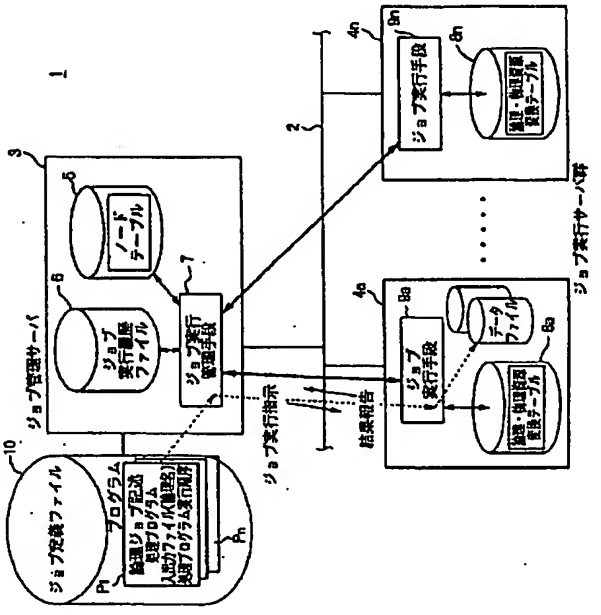
最終頁に続く

(54)【発明の名称】　分散処理システム

(57)【要約】

【課題】　体系的に統一した論理ファイル名によるプログラムの記述を許し、かつ、統一的なジョブ実行履歴を出力することができる分散処理システムを提供する。

【解決手段】　分散処理の全体を管理するジョブ管理サーバ３と、プログラムのジョブを実行するジョブ実行サーバ４ａ…４ｎとを設け、処理プログラムと、処理プログラムの実行順序と、論理名ファイルとによって分散処理すべきプログラムを記述し、ジョブ管理サーバ３に、ジョブ実行サーバの名前等を記録したノードテーブル５と、ジョブ実行履歴ファイル６と、ジョブの実行を管理するジョブ実行管理手段７とを備え、ジョブ実行サーバ４ａ…４ｎに、論理名ファイルを物理資源ファイルに読み換える論理・物理資源変換テーブル８ａ…８ｎと、ジョブを実行するジョブ実行手段９ａ…９ｎとを備えた。



【特許請求の範囲】
【請求項１】複数のコンピュータを接続してプログラムを分散処理する分散処理システムにおいて、
接続されたコンピュータのうちの一つを、プログラムの分散処理の全体を管理するジョブ管理サーバとし、他のコンピュータを、それぞれ前記プログラム中の一連の処理からなるジョブを実行するジョブ実行サーバとし、
前記分散処理すべきプログラムを、それぞれ一定の処理を行う処理プログラムと、前記処理プログラムの実行順序と、論理名による入出力ファイルとからなる論理ジョブ記述によるプログラムとし、
前記ジョブ管理サーバに、ジョブ実行サーバの名前とマシンアドレスを記録したノードテーブルと、各ジョブ実行サーバによるジョブ実行結果を記録するジョブ実行履歴ファイルと、ジョブの実行を各ジョブ実行サーバに分配指示し、それらの処理結果を前記ジョブ実行履歴ファイルに記録するジョブ実行管理手段と、を備え、
前記ジョブ実行サーバに、前記論理名による入出力ファイルを各ジョブ実行サーバの物理資源上のファイルに読み換える論理・物理資源変換テーブルと、ジョブを実行するジョブ実行手段と、を備えたことを特徴とする分散処理システム。
【請求項２】前記ジョブ管理サーバは、前記ジョブ実行履歴ファイルを参照し、各ジョブの実行開始時間および終了時間と、入出力ファイルと、ジョブの実行結果と、を含むジョブリストを出力することを特徴とする請求項１に記載の分散処理システム。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、複数のコンピュータを接続してプログラムを分散処理する分散処理システムに係り、特に処理すべきプログラムについて、実際に接続されているコンピュータの物理的な記憶装置名等を意識せずに、体系的に統一した論理ファイル名による記述を許し、かつ、プログラムの処理結果について、ジョブ実行履歴を出力可能な分散処理システムに関する。
【０００２】
【従来の技術】従来、大きなプログラムの処理は、メインフレームと呼ばれる大型コンピュータによって統一的に処理していた。しかし、システムの柔軟性、経済性、システムの信頼性等の要求から、最近ではミニコンピュータやワークステーションやパーソナルコンピュータ等を通信回線によって接続し、プログラムを分散処理することによって、大型コンピュータと同等の処理を行おうとする要求が高まっていた。
【０００３】この要求に応えて最近では、種々の分散型オペレーティングシステム（分散型ＯＳという）が提案されている。これら分散型ＯＳによれば、統一した文法でプログラムを記述でき、かつ、他のコンピュータ内のファイルにアクセスできるようになった。この分散型ＯＳによって、大型のプログラムを複数のコンピュータによって分散処理することが技術的に可能になった。このような分散型ＯＳとしては、たとえばＵＮＩＸがある。
【０００４】図７は、従来の分散型ＯＳによってプログラムを分散処理する場合のハードウェアのシステム構成を示している。
【０００５】図７に示すように、従来の分散処理システム１１は、複数のコンピュータ１２ａ，…，１２ｎを通信回線１３によって接続したものからなる。
【０００６】各コンピュータ１２ａ，…，１２ｎは、それぞれ異なる物理資源の構成、たとえば、データファイルを格納する記憶装置等を有しているが、同様のジョブ実行手段１４ａ，…，１４ｎによってプログラムを処理する点で共通し、互いに並列的である。
【０００７】上記構成の分散処理システム１１で処理するプログラム１５は、物理資源・環境に基づいて記述されなければならない。すなわち、プログラム１５は、ジョブ（プログラム中の一連の処理）およびコンピュータ１２ａ，…，１２ｎの実行順序を指定し、コンピュータ１２ａ，…，１２ｎのそれぞれの具体的な記憶装置名、ディレクトリー名を指定して入出力ファイルを記述していなければならない。
【０００８】図７の例では、矢印に示すように、プログラム１５は、最初にコンピュータ１２ａで処理され、次にその処理結果がコンピュータ１２ｂに転送されて処理され、さらに、コンピュータ１２ｎで処理されて最終的な処理結果がコンピュータ１２ａに返送される。
【０００９】
【発明が解決しようとする課題】しかしながら、上記従来の分散処理システムでは、プログラムの記述のための作業が煩雑であり、かつ、システム構成の拡張や変更に柔軟に対応するのが困難であった。
【００１０】すなわち、上記分散型ＯＳによるプログラムの記述は、統一した文法によって記述できるものの、プログラム作成者は、システムを構成する各コンピュータの物理的な装置名を念頭に置きながら、データの所在や、エラー発生時の処理等を、それらの物理的な装置によって明示的に記述しなければならなかった。このように、プログラム中の入出力ファイルについてその物理的な所在を一々記述するのは、プログラム作成者にとって大きな負担となっていた。特に、大規模のシステムにおいては、前述したプログラム作成者の負担が著しく増大した。
【００１１】また、このようにして作成したプログラムは、固有のシステム構成にのみ有効であり、異なる構成の分散処理システムに使用することができなかった。したがって、このような分散処理システムでは、システムを構成するコンピュータを追加、変更する場合、あるいは各コンピュータの装置構成を変更する場合、再度プログラムの内容を見直さなければならなかった。これで

は、分散処理システムの長所である柔軟性を十分に活かすことはできなかった。

【００１２】さらに、図７の例から明らかなように、従来の分散処理システムでは、処理結果がシステムのコンピュータ間でやり取りされるので、たとえば、コンピュータ１２ｂの処理に支障があって、処理結果がコンピュータ１２ｎに転送されなかった場合、プログラムのどの部分に支障が発生したかを特定することができなかった。

【００１３】また、たとえすべてのジョブが円満に処理された場合でも、どのコンピュータがどの程度稼働したかのデータを得ることができなかった。

【００１４】特に、複数のプログラムを同時並行的に処理する場合、各コンピュータの負荷を考慮することができなかった。

【００１５】そこで、本発明が解決しようとする課題は、分散型システムにおいて、体系的に統一した論理ファイル名によるプログラムの記述を許し、かつ、大型コンピュータと同等以上の処理結果を得られる分散処理システムを提供することにある。

【００１６】

【課題を解決するための手段】上記課題を解決するために、本願請求項１に係る分散処理システムは、複数のコンピュータを接続してプログラムを分散処理する分散処理システムにおいて、接続されたコンピュータのうちの一つを、プログラムの分散処理の全体を管理するジョブ管理サーバとし、他のコンピュータを、それぞれ前記プログラム中の一連の処理からなるジョブを実行するジョブ実行サーバとし、前記分散処理すべきプログラムを、それぞれ一定の処理を行う処理プログラムと、前記処理プログラムの実行順序と、論理名による入出力ファイルとからなる論理ジョブ記述によるプログラムとし、前記ジョブ管理サーバに、ジョブ実行サーバの名前とマシンアドレスを記録したノードテーブルと、各ジョブ実行サーバによるジョブ実行結果を記録するジョブ実行履歴ファイルと、ジョブの実行を各ジョブ実行サーバに分配指示し、それらの処理結果を前記ジョブ実行履歴ファイルに記録するジョブ実行管理手段と、を備え、前記ジョブ実行サーバに、前記論理名による入出力ファイルを各ジョブ実行サーバの物理資源上のファイルに読み換える論理・物理資源変換テーブルと、ジョブを実行するジョブ実行手段と、を備えたことを特徴とするものである。

【００１７】また、本願請求項２に係る分散処理システムは、上記請求項１の分散処理システムにおいて、前記ジョブ管理サーバは、前記ジョブ実行履歴ファイルを参照し、各ジョブの実行開始時間および終了時間と、入出力ファイルと、ジョブの実行結果と、を含むジョブリストを出力することを特徴とするものである。

【００１８】上記分散処理システムでは、ジョブ管理サーバのジョブ実行管理手段が論理ジョブ記述したプログラムを読み込み、プログラムを構成するジョブの実行を各ジョブ実行サーバに分配・指示する。

【００１９】上記ジョブ管理サーバからジョブの実行を指示されたジョブ実行サーバは、論理・物理資源変換テーブルにより、プログラムに記述された論理名ファイルをそのジョブ実行サーバが有している物理的な装置名上のファイルに読み換えた後に、ジョブ実行手段によって実際にジョブを実行する。

【００２０】ジョブの実行を終了したジョブ実行サーバは、ジョブの実行結果をジョブ管理サーバに報告する。このジョブ実行結果の報告を受けたジョブ管理サーバは、各ジョブの実行結果をジョブ実行履歴ファイルに登録する。

【００２１】さらに、ジョブ管理サーバは、要求により、ジョブ実行履歴ファイルを参照し、ジョブの実行履歴、すなわち、各ジョブの実行開始時間および終了時間、入出力ファイル、各ジョブの実行結果等を含むジョブリストを出力することができる。

【００２２】

【発明の実施の形態】以下に本発明の一実施形態による分散処理システムについて願書に添付の図面を用いて説明する。

【００２３】図１は、本分散処理システムのシステム構成とその処理の流れを示している。図１に示すように、本分散処理システム１は、複数のコンピュータが通信回線２によって接続されたものからなり、接続されたコンピュータの１つを、プログラムの分散処理の全体を管理するジョブ管理サーバ３とし、他のコンピュータを、それぞれプログラム中の一連の処理からなるジョブを実行するジョブ実行サーバ４ａ…４ｎとしたものである。

【００２４】ジョブ管理サーバ３は、ジョブ実行サーバ４ａ…４ｎの名前とマシンアドレスを記録したノードテーブル５と、各ジョブ実行サーバによるジョブ実行結果を記録するジョブ実行履歴ファイル６と、ジョブの実行を各ジョブ実行サーバに分配指示し、管理し、それらの処理結果をジョブ実行履歴ファイル６に記録するジョブ実行管理手段７とを備えている。

【００２５】ジョブ実行サーバ４ａ…４ｎは、論理名による入出力ファイルをそれぞれのジョブ実行サーバの物理資源上のファイルに読み換える論理・物理資源変換テーブル８ａ…８ｎと、実際にジョブを実行し、実行結果をジョブ管理サーバ３に報告するジョブ実行手段９ａ…９ｎとを備えている。

【００２６】この他に、本分散処理システム１では、ジョブ管理サーバ３に分散処理すべきプログラムを供給するジョブ定義ファイル１０を有している。このジョブ定義ファイル１０は、システム内部あるいは外部の所定の記憶装置に記憶されており、一定の処理を行う処理プログラム名と、それら処理プログラムの実行順序と、論理名による入出力ファイルとによって記述した（この記述

方法を本明細書では論理ジョブ記述という）プログラムＰ１ …Ｐn を多数格納している。

【００２７】本分散処理システム１においては、上記論理ジョブ記述されたプログラムＰ１…Ｐn が、ジョブ定義ファイル１０からジョブ管理サーバ３のジョブ実行管理手段７へ送られ、ジョブ実行管理手段７は、プログラムＰ１ …Ｐn の記述に従ってプログラムＰ１ …Ｐn を構成するジョブ（プログラム中の一連の処理）を各ジョブ実行サーバ４ａ…４ｎに分配してその実行を指示する。

【００２８】ジョブ実行サーバ４ａ…４ｎのジョブ実行手段９ａ…９ｎは、論理・物理資源変換テーブル８ａ…８ｎを参照してプログラム中の論理名ファイルをそれぞれの物理資源上のファイルに読み換え、ジョブを実行する。

【００２９】ジョブの処理が終了すると、各ジョブ実行サーバ４ａ…４ｎのジョブ実行手段９ａ…９ｎは、ジョブの実行結果（本明細書では、ジョブ実行結果は、ジョブが支障なく処理されたか否かの信号、ジョブの実行に費やした時間、入出力ファイル等を指す）をジョブ管理サーバ３のジョブ実行管理手段７に報告する。これを受けて、ジョブ実行管理手段７は、各ジョブの実行結果をジョブ実行履歴ファイル６に登録する。ジョブの実行によって得られたデータ（これを本明細書では「ジョブ実行結果」と区別して「処理データ」という）は、各ジョブ実行サーバ４ａ…４ｎからジョブ管理サーバ３へ送られ、図示しない出力手段を介して出力される。

【００３０】次に、本分散処理システム１によって、分散処理すべきプログラムＰ１ …Ｐnの記述が如何に簡略化され、かつ、拡張・柔軟性を備えて大型のメインフレームと同等の処理を実現するかをプログラムＰ１ …Ｐn 等の具体例を示して以下に説明する。

【００３１】図２は、本分散処理システム１に使用される論理ジョブ記述されたプログラムの一例と、その各センテンスについての説明を示している。図３は、図２のプログラムの処理フローを概念的に示したフロー図である。

【００３２】図２に示すように、この論理記述されたプログラムは、a.jcl （１行目～１６行目）と、b.jcl （１７行目～４１行目）と、c.jcl （４２行目～５０行目）の３つのジョブについて記述している。各ジョブは、「／」によって区切られた「ステップ」という処理単位の組合せからなる。

【００３３】今、「a.jcl 」というジョブについて注目すると、最初にジョブ名（１行目）と、ＰＧＭ（プログラムの略）のロードモジュールの格納場所のサーチ順序（２行目）と、ジョブの実行優先度（３行目）とについて記述している。

【００３４】次に、ステップ（４行目～１０行目）の処理について記述している。このステップの処理では、最初にステップ名を記述し（５行目）、次に、一定の処理を行う処理プログラム「pgm1」の実行を指示し（６行目）、次に２行目サーチ順序の下位のプログラムの格納場所を指示した後に（７行目）、入力するストレージファイル「SYS10 se.ia1」と（８行目）、出力する一時ファイル「SYS20 &;ia1」を指示している。ここで、「SYS10 」や「SYS20 」は、入力や出力ファイルを示す一般的な取り決めであり、「se.ia1」や「&;ia1」が各ファイルの固有名称である。

【００３５】上記ステップの意味は、ストレージファイル「se.ia1」を入力し、処理プログラム「pgm1」によってストレージファイル「se.ia1」を処理し、処理結果を一時ファイル「&;ia1」に出力せよ、というものである（図３参照）。

【００３６】次のステップ（１０行目～１５行目）では、プログラム「pgm2」の実行を指示し（１２行目）、先に出力した一時ファイル「&;ia1」を入力し（１３行目）、プログラム「pgm2」によって一時ファイル「&;ia1」を処理した結果をシステム共通の固定ファイル「/fix/se.iam 」に書き込む（１４行目）をことを指示している。上記２つのステップによってジョブ「a.jcl 」の処理は終了する。

【００３７】同様に、「b.jcl 」は、図３に理解容易に示すように、固定ファイル「/fix/se.iam 」とストレージファイル「se.ia2」を入力し、プログラム「pgm3」によって処理し、処理結果をデータごとに、メモリ上でプログラム「pgm4」に受け渡し（これをパイプ機能という）、プログラム「pgm4」によって処理し、処理結果を一時ファイル「&;ia3」に書き込む。次に、一時ファイル「&;ia3」と、固定ファイル「/fix/se.iam 」とを読み込み、プログラム「pgm5」によって処理し、処理結果をストレージファイル「se.ia3」に書き込む。さらに、このストレージファイル「se.ia3」をノードＢのストレージファイル「se.ia3」に転送する。ここで、ノードＢとは、分散処理システムに接続されている特定のコンピュータを示しており、実際的には特定のジョブ実行サーバの名称を記入する。

【００３８】以上でジョブ「b.jcl 」の処理は終了する。なお、上記ジョブ「a.jcl 」、「b.jcl 」は、プログラムの最初に書かれているため、無条件にノードＡによって処理されるが、次のジョブ「c.jcl 」からは、指定されたノードＢで実行される。

【００３９】ジョブ「c.jcl 」は、ストレージファイル「se.ia3」を入力し、このファイルをプログラム「pgm6」によって処理し、処理結果をストレージファイル「se.ia4」に書き込むことを指示している。

【００４０】上記プログラムの記述上の特徴について以下に説明する。本分散処理システム１で処理するプログラムは、「se.ia1」，…「se.ia4」（ストレージファイル）、「&;ia1」，…，「&;ia3」（一時ファイル）、「#ia2」（パイプファイル）のように、予め体系的に定めた

論理ファイル名を用いている。

【００４１】このことが、従来のプログラムの記述と大きな差異をなす。すなわち、従来、上記入出力ファイル、特に「一時ファイル」「パイプファイル」等は、プログラム作成者が実際にジョブを実行するコンピュータ（ここではノードＡとノードＢ）の物理資源の構成を考慮し、各コンピュータの特定の記憶装置名の特定のディレクトリー（ファイルの格納場所）名を具体的に明記しなければならなかった。

【００４２】したがって、ファイルごとに各ジョブ実行サーバごとに異なる記憶装置名やディレクトリー名を調べなければならず、このことがプログラム作成上の大きな負担となっていた。ノード名（ジョブ実行サーバの名前、マシンアドレス）についても同様であった。

【００４３】これに対して本分散処理システム１で処理するプログラムは、最初にファイルについて「ストレージファイル」、「一時ファイル」、「パイプファイル」に分け、各種類のファイルごとに所定の名称とシリアルナンバーを付したファイル名の体系を定めておけば、プログラム作成者は、ジョブ実行サーバの物理環境を意識せずにプログラムを作成できる。

【００４４】上記プログラムがジョブ管理サーバ３のジョブ実行管理手段７に読み込まれたときは、ジョブ実行管理手段７は、ノードテーブル５を参照して、プログラムに記述したノード名に従って各ジョブを各ジョブ実行サーバ４ａ…４ｎに分配し、実行を指示する。

【００４５】ここで、ノードテーブル５の一例を図４に示す。図４に示すように、ノードテーブル５は、各ジョブ実行サーバ４ａ…４ｎのマシンアドレスと装置名を列記したものである。このノードテーブル５により、プログラムに記述されたノード名を検索し、各ジョブ実行サーバを特定することができる。

【００４６】ジョブ実行管理手段７からのジョブ実行命令を受けたジョブ実行サーバ４ａ…４ｎは、論理・物理資源変換テーブル８ａ…８ｎを参照して、論理名ファイルを各ジョブ実行サーバの物理的な装置上のファイルに読み換え、ジョブを実行する。

【００４７】ここで論理・物理資源変換テーブル８ａ…８ｎの一例を図５に示す。図５の６行目～８行目は、ストレージファイルを格納する物理的な装置名あるいはディレクトリー名を示している。同様に、図５の９行目と１０行目は一時ファイルを格納する物理的装置名・ディレクトリー名、同１１行目はパイプファイルを格納する物理的装置名・ディレクトリー名をそれぞれ示している。

【００４８】上記論理・物理資源変換テーブル８ａ…８ｎにより、ジョブ実行サーバ４ａ…４ｎは、それぞれの物理装置上でファイルを扱えることができ、支障なくジョブを実行することができる。

【００４９】このようにしてジョブを実行したジョブ実行サーバ４ａ…４ｎは、ジョブの実行結果をジョブ実行管理手段７に報告し、ジョブ実行管理手段７はこれらのジョブの実行結果をジョブ実行履歴ファイル６に登録する。

【００５０】このジョブ実行履歴ファイル６を参照することにより、本発明による分散処理システム１は、メインフレーム等の大型コンピュータ同様のジョブリストを出力することができる。

【００５１】ジョブリストは、各ジョブの実行開始時間および終了時間と、入出力ファイルと、ジョブの実行結果とを含み、プログラムの検証に大きく役立つことができる。

【００５２】図６は、ジョブリストの一例を示している。図６に示すように、ジョブリストを参照することにより、各ジョブ（このリストではSEIA000, SEIA100 等）の実行開始時間、実行終了時間、入出力ファイル、ＣＰＵの使用時間等の情報を得ることができる。万一、プログラムの不備等により処理が中断した場合は、どのジョブの実行によって処理が中断したかを特定でき、プログラムの誤りの発見に役立てることができる。

【００５３】本発明の分散処理システムによれば、プログラムは処理の思想（抽象化した処理の流れ）のみを示しており、システムに接続されたコンピュータの物理環境に影響されない。すなわち、たとえば、分散処理システムにジョブ実行サーバを追加したり、あるいは所定のジョブ実行サーバの記憶装置の構成を変更するような場合、プログラムを書き直すことなく、ノードテーブルや論理・物理資源変換テーブルの一部の変更によって対応することができるのである。

【００５４】また、上記説明では、一つのプログラムを処理する場合について説明したが、本発明の分散処理システムは、一つのプログラムの処理に限らず、大型コンピュータ同様に複数のプログラムを同時並行的に処理することができる。

【００５５】この場合、ジョブ実行管理手段７は、プログラムＰ１ …Ｐｎ に従って、どのプログラムに属するジョブであるかに関係なく、ジョブの実行順序を付して次々に各ジョブ実行サーバ４ａ…４ｎにジョブの実行を指示する。

【００５６】各ジョブ実行サーバ４ａ…４ｎでは、未処理のジョブは待ちの状態になり、実行順序に従ってジョブが逐次処理される。処理されたジョブはジョブ管理サーバ３に報告され、ジョブ管理サーバ３は、プログラムごとに処理結果、処理データを出力する。

【００５７】

【発明の効果】上記説明から明らかなように、本発明の分散処理システムによれば、接続されたコンピュータの物理資源を意識せずに、論理ファイル名によってプログラムを記述することができ、プログラム作成上の負担が大幅に軽減することができる。

【００５８】また、ノードテーブルや論理・物理資源変換テーブル等の局部的な修正をすることにより、システムの構成を拡張・変更することができる。

【００５９】以上により、本発明によれば、プログラムの作成が容易であり、かつ、システムの拡張・変更に柔軟に対応でき、さらに、小型コンピュータを接続した環境下で、大型コンピュータと同様の処理を実現する分散処理システムを提供することができる。

【図面の簡単な説明】

【図１】本発明による分散処理システムのシステム構成と処理の流れを示したブロック図。

【図２】本発明による分散処理システムで処理するプログラムの一例とその説明文を併記した説明図。

【図３】図２に示したプログラムの処理の流れを概念的に示したフロー図。

【図４】ノードテーブルの一例を示した説明図。

【図５】論理・物理資源変換テーブルの一例を示した説明図。

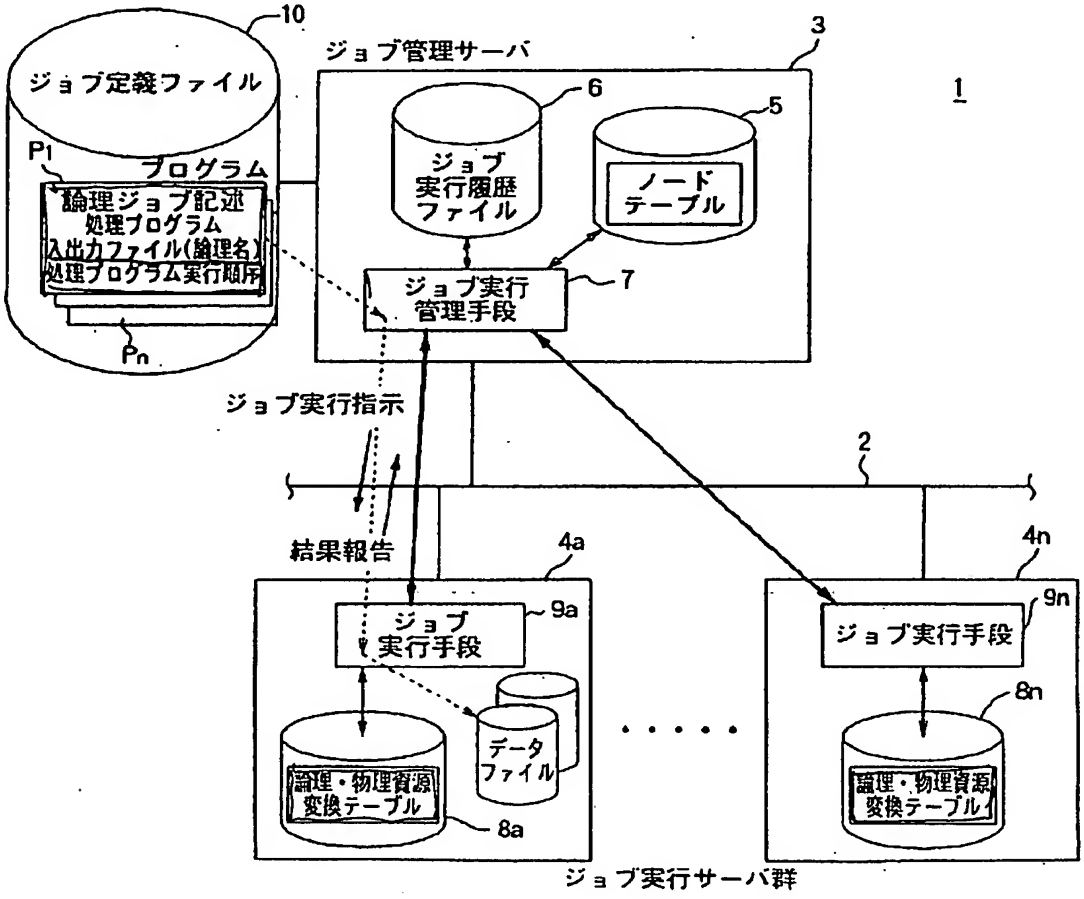
【図６】ジョブリストの一例を示した説明図。

【図７】従来の分散処理システムのシステム構成と処理の流れを示したブロック図。

【符号の説明】

１　分散処理システム
２　通信回線
３　ジョブ管理サーバ
４　ジョブ実行サーバ
５　ノードテーブル
６　ジョブ実行履歴ファイル
７　ジョブ実行管理手段
８　論理・物理資源変換テーブル
９　ジョブ実行手段
１０　ジョブ定義ファイル
Ｐ　プログラム

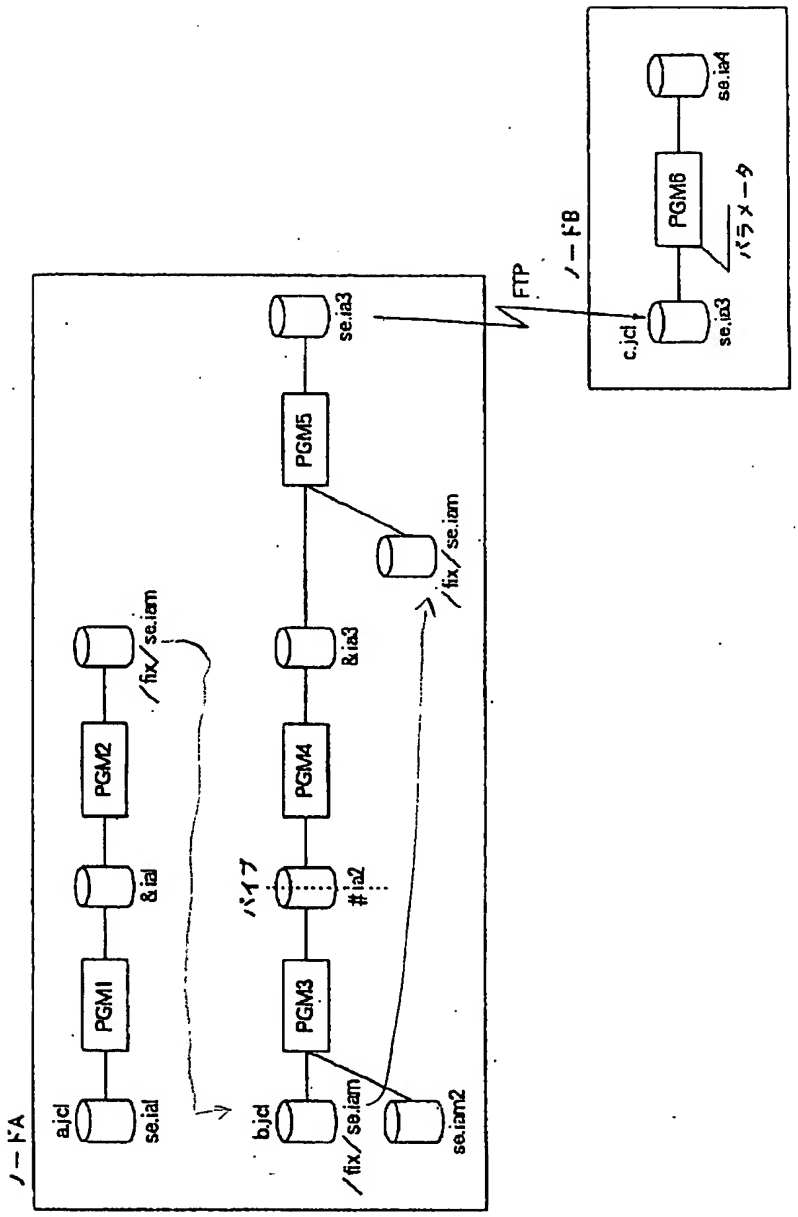
【図１】



【図２】

行	プログラム	説　明
1	* a.jcl	コメント行（ＪＯＢ名記述）
2	JL /ris/bin /test/bin	ＪＯＢ中のＰＧＭのロードモジュールをサーチする順序
3	JP n	ＪＯＢ実行中のプライオリティ
4	/	ステップ間区切り記号
5	SN PGM1	ステップ名記述（ログを取るため）
6	S pgm1	ＰＧＭの実行
7	SL /users/yamawaki	該当ステップのＰＧＭのロードモジュール格納場所
8	I SYS010 se.ia1	ストレージファイルの読込
9	O SYS020 &ia1	一時ファイルへの書込
10	/	ステップ間区切り記号（／から／までを一度に処理）
11	SN PGM2	
12	S pgm2	
13	I SYS010 * &ia1.DELETE	一時ファイルの読込、ステップ終了後に消去
14	O SYS020 /fix/se.iam	固定ファイルへの書込
15	/	
16	/	ＪＯＢの終了（一時ファイル等のクリア）
17	* b.jcl	
18	JL /ris/bin	
19	/	
20	SN PGM3	
21	S pgm3	
22	I SYS010 /fix/se.iam	固定ファイルの読込
23	I SYS011 se.ia2	ストレージファイルの読込
24	O SYS020 #ia2	パイプファイルへの書込
25	&	パイプ接続
26	SN PGM4	
27	S pgm4	
28	I SYS010 #ia2	パイプファイルからの読込
29	O SYS020 &ia3	一時ファイルへの書込
30	/	
31	SN PGM5	
32	S pgm5	
33	I SYS010 &ia3	一時ファイルの読込
34	I SYS011 /fix/se.iam	
35	O SYS020 se.ia3	
36	/	
37	SN PTP	
38	S ftp　ノードB	ｆｔｐの起動、相手先ノード名を指定
39	I SYS010 se.ia3	相手先でのファイル名を指定
40	/	
41	/	
42	* c.jcl	
43	JL /ris/bin	
44	/	
45	SN PGM6	
46	S pgm6 パラメータ	プログラムのパラメータ記述
47	I SYS010 se.ia3	
48	O SYS020 se.ia4	
49	/	
50	/	

【図3】


ノードA
a.jcl
se.ial
PGM1
&ial
PGM2
/fix/se.iam
b.jcl
/fix/se.iam
PGM3
パイプ
#ia2
PGM4
&ia3
PGM5
se.ia3
/fix/se.iam
se.iam2
FTP
ノードB
c.jcl
se.ia3
PGM6
パラメータ
se.ia4

【図4】

```
1   Session Name: yokes1

2   nobu@yokes1[/unyo/config]:$ more nodefile
3   HP0     csshp0
4   HP1     hockey
5   KUN1    yokun1
6   JUN1    opjun1
7   KES1    yokes1
8   yokes1  yokes1
9   KES2    yokes2
10  yokes2  yokes2
```

【図5】

```
1   Session Name: yokes1
            ..............
            ..............
2   devfile
            ..............
            ..............
3   /dev/rmt/1h DEV_1 220
4   /dev/rmt/2h DEV_2 220
            ..............
            ..............
5   dirfile
            ..............
            ..............
6   s/data1/strg
7   s/data2/strg
8   s/data3/strg
9   t/data1/temp
10  t/data2/temp
11  f/data1/fifo
            ..............
            ..............
12  srtfile
            ..............
            ..............
13  /data3/temp
```

se.ia1 → 6 → s/data1/strg
7 → s/data2/strg
8 → s/data3/strg
ストレージファイル

&ia1 → 9 → t/data1/temp
&ia3 → 10 → t/data2/temp
一時ファイル

#ia2 → 11 → f/data1/fifo
パイプファイル

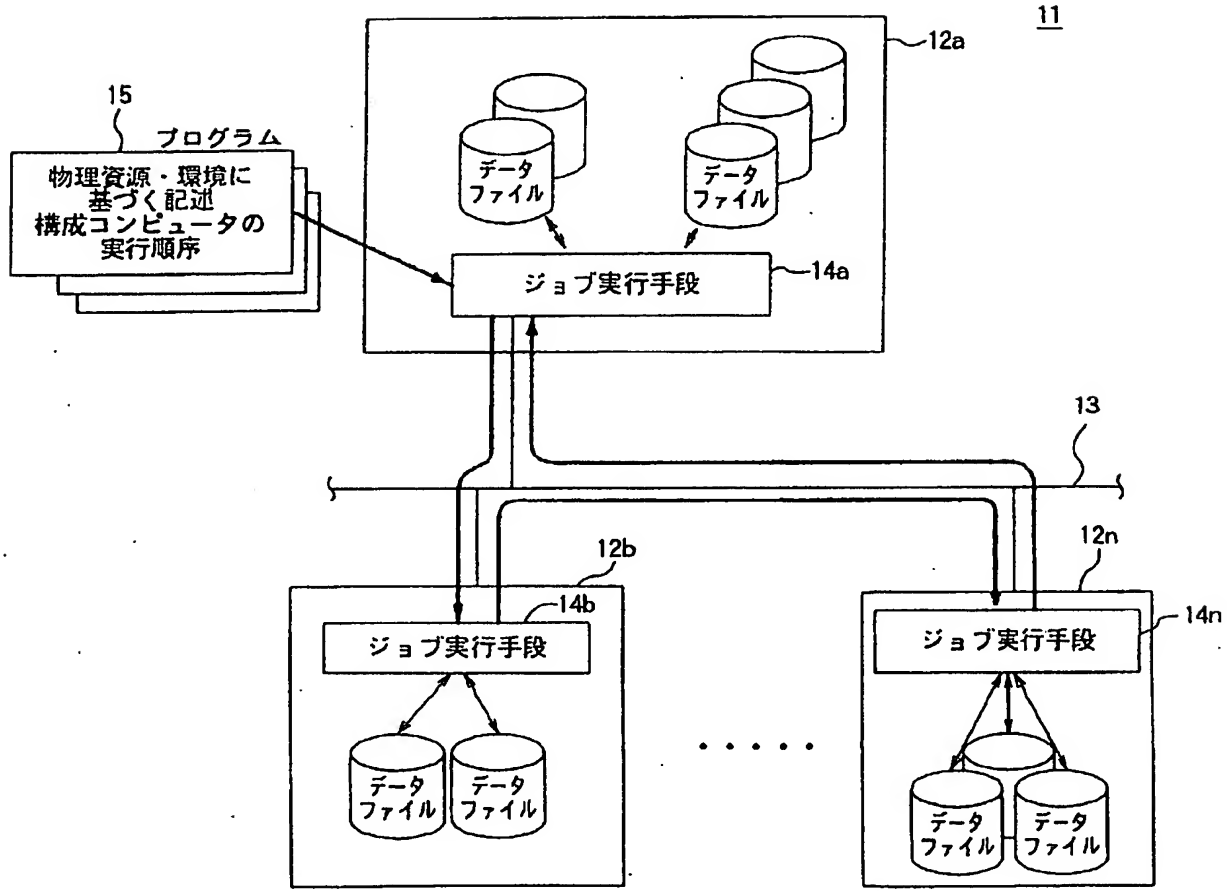
13 /data3/temp
一時ファイル

【図６】

```
############ E X E C  L O G ############

22556 19:04:55 SETEST JOB STARTED CLASS=B NODE=hpt
      19:04:55 STEP SEIA000 STARTED
                    SEIA000 SYS010=/ryu1/unyo/data/strg1/SE.HCODE
                    SEIA000 SYS020=/ryu1/unyo/data/strg2/SE.IA00
                    SEIA000 CPU TIME: 0.010000
      19:04:55 STEP SEIA000 ENDED CODE=0
      19:04:56 STEP SEIA100 STARTED
                    SEIA100 SYS010=/ryu1/unyo/data/strg1/SE.IAMAAE
                    SEIA100 SYSHDD=/ryu1/unyo/data/strg2/SE.IA00
                    SEIA100 SYS020=/ryu1/unyo/data/temp3/IA100_26623
                    SEIA100 CPU TIME: 0.050000
      19:04:56 STEP SEIA100 ENDED CODE=0
      19:04:56 STEP SEIA102S STARTED
                    SEIA102S SORTIN=/ryu1/unyo/data/temp3/IA100_26623
                    SEIA102S SORTOUT=/ryu1/unyo/data/temp1/IA102S_26623
                    SEIA102S CPU TIME: 0.040000
      19:04:57 STEP SEIA102S ENDED CODE=0
      19:04:57 STEP SEIA102 STARTED
                    SEIA102 SYS010=/ryu1/unyo/data/temp1/IA102S_26623
                    SEIA102 SYS020=/ryu1/unyo/data/strg3/SE.IAMISED
                    SEIA102 CPU TIME: 0.040000
      19:04:57 STEP SEIA102 ENDED CODE=0
               SETEST REAL TIME: 2.540000
               SETEST SCPU TIME: 1.070000
               SETEST UCPU TIME: 0.290000
22556 19:04:57 SETEST JOB ENDED CODE=0
```

【図７】



フロントページの続き

(72)発明者　吉　田　倡　克
神奈川県横浜市保土ケ谷区神戸町134番地
株式会社野村総合研究所内

(72)発明者　稲　田　陽　一
神奈川県横浜市保土ケ谷区神戸町134番地
株式会社野村総合研究所内

(72)発明者　山　脇　昌　則
神奈川県横浜市保土ケ谷区神戸町134番地
株式会社野村総合研究所内

(72)発明者　藤　島　美　保
神奈川県横浜市保土ケ谷区神戸町134番地
株式会社野村総合研究所内

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 11-085694
(43)Date of publication of application : 30.03.1999

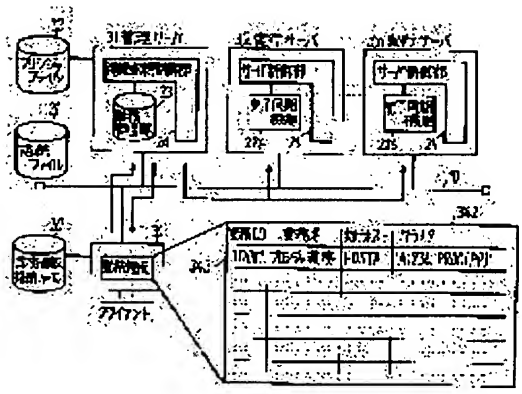
(51)Int.Cl. G06F 15/00
G06F 13/00

(21)Application number : 09-246918
(22)Date of filing : 11.09.1997

(71)Applicant : FUJITSU LTD
(72)Inventor : SUENARI TORU

(54) INTER-SERVER LINK JOB OPERATING SYSTEM

(57)Abstract:
PROBLEM TO BE SOLVED: To request a job later from a specified managing server to respective execution servers while decomposing the execution requests of respective jobs only by simply requesting continuous procedures to this managing server without requesting the execution of respective jobs from a client, which requests the plural continuous procedures, to the execution servers for every processing.
SOLUTION: In a computer system connected to respective lines 10, corresponding to the plural continuous procedures from the client, a managing server 31 determines the analysis of respective procedures and the execution procedure at the request destination and after the job control sentence of the leading job is generated, the execution is requested to execution servers 32-3n. At the execution server, the end of the requested server job is waited, the succession information or output file to the next server job are returned to the managing server 31. Based on this result, the managing server 31 generates the next job control sentence and requests the execution to the next execution server. By repeating this operation, the managing server 31 completes the request from the client.



LEGAL STATUS

[Date of request for examination]	28.05.2003
[Date of sending the examiner's decision of rejection]	31.10.2006
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]	
[Date of final disposal for application]	
[Patent number]	
[Date of registration]	
[Number of appeal against examiner's decision of rejection]	2006-027039
[Date of requesting appeal against examiner's decision of rejection]	30.11.2006
[Date of extinction of right]	

1 / 1